Learning stochastic eigenvalues

Roman Andreev

▶ **To cite this version:**

Roman Andreev. Learning stochastic eigenvalues. 2016. hal-01313404v2

HAL Id: hal-01313404
https://hal.science/hal-01313404v2

Preprint submitted on 12 Dec 2016

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Distributed under a Creative Commons Attribution| 4.0 International License

LEARNING STOCHASTIC EIGENVALUES

ROMAN ANDREEV†

Abstract. We train an artificial neural network with one hidden layer on realizations of the first few eigenvalues of a partial differential operator that is parameterized by a vector of independent random variables. The eigenvalues exhibit "crossings" in the high-dimensional parameter space. The training set is constructed by sampling the parameter either at random nodes or at the Smolyak collocation nodes. The performance of the neural network is evaluated empirically on a large random test set. We find that training on random or quasi-random nodes is preferable to the Smolyak nodes. The neural network outperforms the Smolyak interpolation in terms of error bias and variance on nonsimple eigenvalues but not on the simple ones.

1. The stochastic eigenvalue problem

In this note we compare the out-of-sample prediction of the Smolyak sparse grid interpolant with that of a simple artificial neural network for a stochastic eigenvalue problem. Our model problem is the parametric partial differential eigenvalue problem

$$-\nabla\cdot(c(x,y)\nabla u(x,y))+u(x,y)=\lambda(y)u(x,y),\qquad x\in D,\quad y\in Y, \tag{1}$$

where $D:=(-1,1)^2$ is the spatial domain and $Y:=[0,1]^M$ is the parameter space. We impose homogeneous Neumann spatial boundary conditions. The conductivity coefficient has the form

$$c(x,y):=1+\textstyle\sum_{m=1}^{M}\psi_m(x)y_m. \tag{2}$$

We think of the $y_m\in[0,1]$ as independent uniformly distributed random variables. Throughout, the stochastic dimension is $M=16$ and the ψ_m are the indicator functions of the subdomains obtained by partitioning the spatial domain into 4×4 equal parts. The problem (1) will be discretized with P1 finite elements with 42^2 unknowns using the Matlab PDE toolbox.

Our quantity of interest is the set of the first 9 eigenvalues,

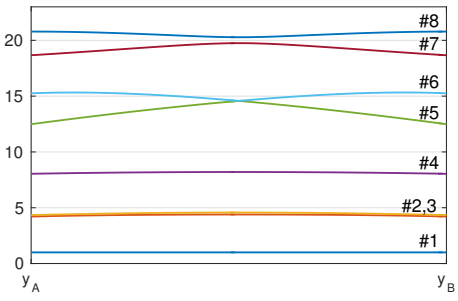
$$\mathsf{QI}(y):=\{1=\lambda_1(y)\le\lambda_2(y)\le\ldots\le\lambda_9(y)\}. \tag{3}$$

For $y=0$, the nontrivial ones are

$$\lambda_{2,3}=\tfrac{0+1}{4}\pi^2+1,\quad \lambda_4=\tfrac{1+1}{4}\pi^2+1,\quad \lambda_{5,6}=\tfrac{0+4}{4}\pi^2+1,\quad \lambda_{7,8}=\tfrac{1+4}{4}\pi^2+1,\quad \lambda_9=\tfrac{4+4}{4}\pi^2+1,$$

and the last one is simple.

However, as we vary the parameter, the eigenvalues split, leading to a nonsmooth (Lipschitz) dependence on the parameter. For example (figure on the right and Figure 1), interpolating linearly between $c(\cdot,y_A)=1+\mathbb{1}_{x_1>0}$ and $c(\cdot,y_B)=1+\mathbb{1}_{x_2>0}$, the eigenvalues #2–3 remain almost identical (slight discretization asymmetry), #5–6 switch midway, while #7–8 (and also #9) remain distinct. These eigenvalue "crossings" or "switchings" make it difficult to capture the parameter dependence over the whole high-dimensional parameter space.



In the next section we discuss two regression methods for that purpose, both based on sampling in the parameter domain: Smolyak interpolation and artificial neural networks.

Date: December 12, 2016.

2010 *Mathematics Subject Classification.* 15B51, 35R60, 62M45, 62M40, 65N25, 65N30, 65Y05, 65Y20, 68T37.

Key words and phrases. uncertainty quantification, stochastic eigenvalues, neural networks, sparse grids.

2. Two regression methods

2.1. **The Smolyak interpolant** [2]**.** For each integer level $\ell \geq 0$ let i_ℓ be a univariate interpolation operator on the interval $[0,1]$. We will use the polynomial interpolation operator based on the Clenshaw–Curtis nodes

$$N_0 = \{\tfrac{1}{2}\} \quad \text{and} \quad N_\ell = \{\tfrac{1}{2}(1-\cos(2^{-\ell}k\pi)) : 0 \leq k \leq 2^\ell\}, \quad \ell \geq 1. \tag{4}$$

For a multiindex set $\Lambda \subset \mathbb{N}_0^M$, the multivariate Smolyak interpolation operator I_Λ is defined by

$$\mathsf{I}_\Lambda := \textstyle\sum_{\nu\in\Lambda}(j_{\nu_1} \otimes j_{\nu_2} \otimes \ldots \otimes j_{\nu_M}), \tag{5}$$

where $j_\ell := i_\ell - i_{\ell-1}$ is the univariate increment and $i_{-1} := 0$. We confine the discussion to the multiindex sets of a given total level $L \geq 0$,

$$\Lambda_L := \{\nu \in \mathbb{N}_0^M : \textstyle\sum_m \nu_m \leq L\}, \tag{6}$$

and write I_L for the resulting multivariate Smolyak interpolation operator. This multiindex set is monotone: if $\mu \in \Lambda_L$ and if $\nu \leq \mu$ coordinatewise then also $\nu \in \Lambda_L$. This, and the nestedness property $N_{\ell-1} \subset N_\ell$ of the Clenshaw–Curtis nodes imply the unisolvency of the multivariate interpolation operator on the sparse grid (observe the redundancy in the union)

$$\mathcal{N}_L := \textstyle\bigcup_{\nu\in\Lambda_L}(N_{\nu_1} \times N_{\nu_2} \times \ldots \times N_{\nu_M}) \subset Y, \tag{7}$$

because the number of collocation nodes in (7) matches the dimension of the range of the interpolation operator (5). With $M = 16$ stochastic dimensions, the number of collocation nodes is $\#\mathcal{N}_0 = 1$, $\#\mathcal{N}_1 = 33$, $\#\mathcal{N}_2 = 545$, $\#\mathcal{N}_3 = 6'049$, etc. We write

$$\mathsf{Sm}[L] := \mathsf{I}_L \circ \mathsf{QI} : \mathbb{R}^M \to \mathbb{R}^9 \tag{8}$$

for the interpolated quantity of interest.

2.2. **Artificial neural networks.** We consider the simple class of neural networks of the form

$$\text{input } (M \text{ parameters}) \overset{\text{linear}}{\longrightarrow} \text{ReLU } (H \text{ hidden units}) \overset{\text{linear}}{\longrightarrow} \text{output } (9 \text{ real values}), \tag{9}$$

or as the composition

$$\mathsf{NN} : \mathbb{R}^M \to \mathbb{R}^9, \quad \mathsf{NN} = (\mathbf{h} \mapsto \mathbf{Ch} + \mathbf{c}) \circ \mathrm{ReLU} \circ (\mathbf{y} \mapsto \mathbf{By} + \mathbf{b}), \tag{10}$$

with the $(9(H+1) + H(M+1))$ learnable parameters

$$\{\mathbf{C} \in \mathbb{R}^{9\times H}, \mathbf{c} \in \mathbb{R}^9\} = \text{2nd all-to-all layer}, \quad \{\mathbf{B} \in \mathbb{R}^{H\times M}, \mathbf{b} \in \mathbb{R}^H\} = \text{1st all-to-all layer}. \tag{11}$$

The "rectified linear unit" is the function $\mathrm{ReLU} : x \mapsto \max\{x, 0\}$, acting componentwise when applied to a vector. The number of "hidden units" will be $H \in \{M, 4M, 16M\}$.

A training set or a test set is a collection of reference nodes $\mathcal{N} \subset Y$ together with the precomputed values of the quantity of interest (3) at those nodes. We refer to a pair $(y, \mathsf{QI}(y))$ as a sample. We consider two types of reference nodes for training the neural network:

(1) the Smolyak collocation nodes $\mathcal{N}_L$ from (7), and
(2) randomly generated nodes $\widetilde{\mathcal{N}}_L$ of the same cardinality as $\mathcal{N}_L$.

The random training set is generated only once for each level L and used for all configurations. By "average" we will mean the ensemble average over the training set or over the test set.

The learnable parameters are initialized uniformly at random from $[-s, s]$ where $s = 1/\sqrt{M}$ for the first linear layer and $s = 1/\sqrt{H}$ for the second one. Then, they are trained in 10k iterations of stochastic gradient descent. Each iteration is a pass over the training set in random order; for each sample (y, q), the learnable parameters α are adjusted (all at once) by the gradient scheme

$$\alpha \leftarrow \alpha - r\tfrac{\partial}{\partial\alpha}\mathsf{D}(q, \mathsf{NN}(y)), \quad \mathsf{D}(q,p) := \tfrac{1}{9}|q-p|_1, \tag{12}$$

where $r > 0$ is the learning rate (initially, $r = 0.1$) and $|\cdot|_1$ is the vector 1-norm measuring the prediction discrepancy. The derivatives of ReLU and $|\cdot|_1$ are right-continuous. After each iteration, the learning rate is set to 1% of the average prediction discrepancy. The input/output data are offset by their average during the training phase. We write $\mathsf{NN}[H, \mathcal{N}]$ for the neural network with H hidden units trained on the node set $\mathcal{N}$. We use the `torch7` environment for the training.

3. Evaluation

Given a predictor $\mathsf{P} = (8)$ or $\mathsf{P} = (10)$, we evaluate its quality over a fixed test set of 100k samples on random test nodes $\mathcal{N}_{\text{test}} \subset Y$. The average and the standard deviation of the eigenvalues in the test set are as follows (rounded to two postcomma digits):

eigenvalue:	#1	#2	#3	#4	#5	#6	#7	#8	#9
average	1	4.55	4.68	8.24	5.24	15.47	18.77	19.62	29.6
std dev	0	0.2	0.2	0.39	0.74	0.73	0.97	0.98	1.47

Predictions along a non-axiparallel segment in the parameter space are shown in Figure 1.

We examine the errors

$$\delta_k(y) := (\mathsf{QI}(y) - \mathsf{P}(y))_k, \quad y \in \mathcal{N}_{\text{test}}, \tag{13}$$

in the prediction of the k-th eigenvalue. Unsorted predictions occurred at the following rates:

	$\mathsf{Sm}[L]$	$\mathsf{NN}[M, \mathcal{N}_L]$	$\mathsf{NN}[4M, \mathcal{N}_L]$	$\mathsf{NN}[M, \widetilde{\mathcal{N}}_L]$	$\mathsf{NN}[4M, \widetilde{\mathcal{N}}_L]$
$L = 2$	71.5%	3%	0.6%	0.31%	1.3%
$L = 3$	63.5%	2.8%	0.15%	0.02%	0.01%

We call the average $\bar{\delta}_k$ of δ_k the prediction bias. Figures 2–3 show the empirical cumulative distribution function of the bias-free prediction error $(\delta_k - \bar{\delta}_k)$ along with the prediction bias $\bar{\delta}_k$ for different predictors. We observe the following (recall, $M = 16$ is the stochastic dimension):

(1) For the nonsimple eigenvalues $\#2, 3, 5, 6, 7, 8$, the neural network $\mathsf{NN}[4M, \widetilde{\mathcal{N}}_L]$ trained on random nodes performs consistently better in terms of the prediction bias and prediction error variance than the Smolyak interpolation operator $\mathsf{Sm}[L]$.
(2) This behavior is reversed for the simple eigenvalues #4 and #9. This is explained by the analytic dependence of simple eigenvalues on the parameter [1]. Increasing the number of hidden units from $H = 4M$ to $H = 16M$ is still not sufficient to surpass interpolation.
(3) Training the neural network on the Smolyak nodes as opposed to random nodes clearly decreases the prediction accuracy.
(4) The neural network with M hidden units instead of $4M$ is less competitive on training level $L = 3$ than on $L = 2$.
(5) The results are essentially the same for the quasi-random Sobol points (as generated by the Matlab command `sobolset(M)`) instead of the random ones (not shown).

Finally, we note that neural networks and sparse grid interpolation have different distributions of offline/online costs. While the offline training of the neural network is lengthy, the online prediction is much quicker than sparse grid interpolation, especially for high-dimensional parameters. However, both effects can be offset by parallelization.

Acknowledgment

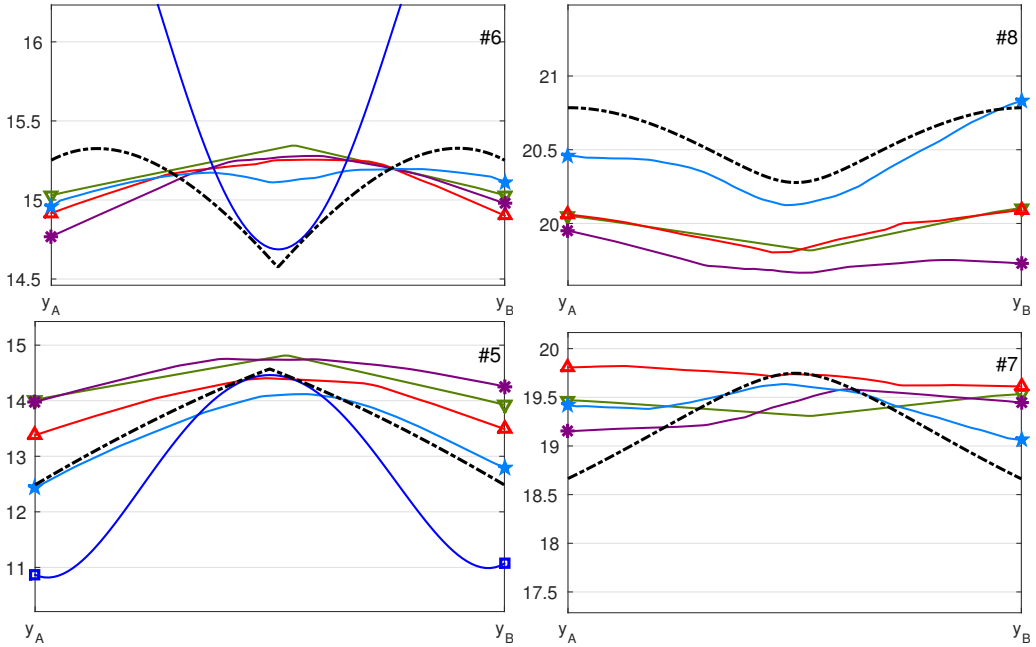
Supported by the Swiss NSF Advanced Postdoc.Mobility grant #164616.

References

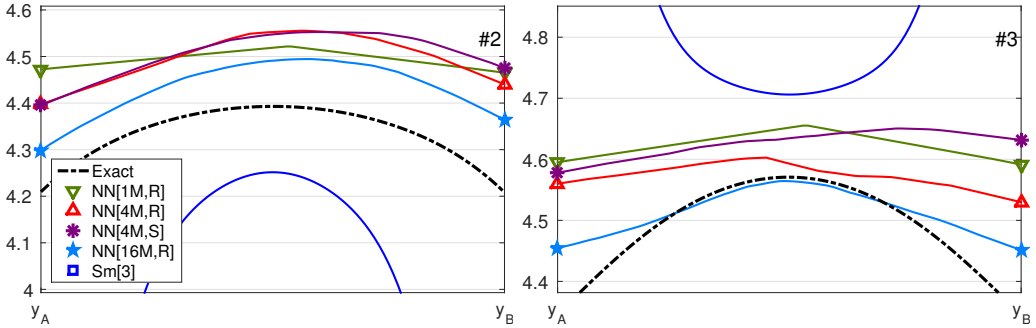
[1] Roman Andreev and Christoph Schwab. Sparse tensor approximation of parametric eigenvalue problems. In Ivan G. Graham, Thomas Y. Hou, Omar Lakkis, and Robert Scheichl, editors, *Numerical Analysis of Multiscale Problems*, volume 83 of *Lecture Notes in CSE*, pages 203–241. Springer Berlin Heidelberg, 2012. 3

[2] Sergey A. Smolyak. Quadrature and interpolation formulas for tensor products of certain classes of functions. *Sov. Math. Dokl.*, 4:240–243, 1963. 2

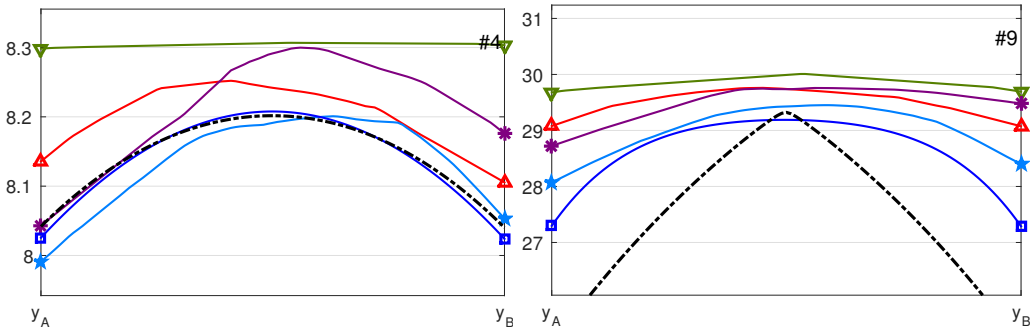
†Université Paris Diderot, Sorbonne Paris Cité, LJLL (UMR 7598 CNRS), F-75205, Paris, France
E-mail address: roman.andreev@upmc.fr



Above: The eigenvalues #5-6 switch (left). The eigenvalues #7-8 come close (right). The Smolyak interpolant for #7-8 is outside the range shown.



Above: The eigenvalues #2-3 are slightly distinct (due to discretization asymmetry).



Above: The simple eigenvalues #4 and #9. Eigenvalue #9 comes close to #10 midway.

FIGURE 1. Exact and predicted eigenvalues along the parameter space segment from $c(\cdot, y_A) = 1 + \mathbb{1}_{x_1>0}$ to $c(\cdot, y_B) = 1 + \mathbb{1}_{x_2>0}$. See #2 for the annotation.

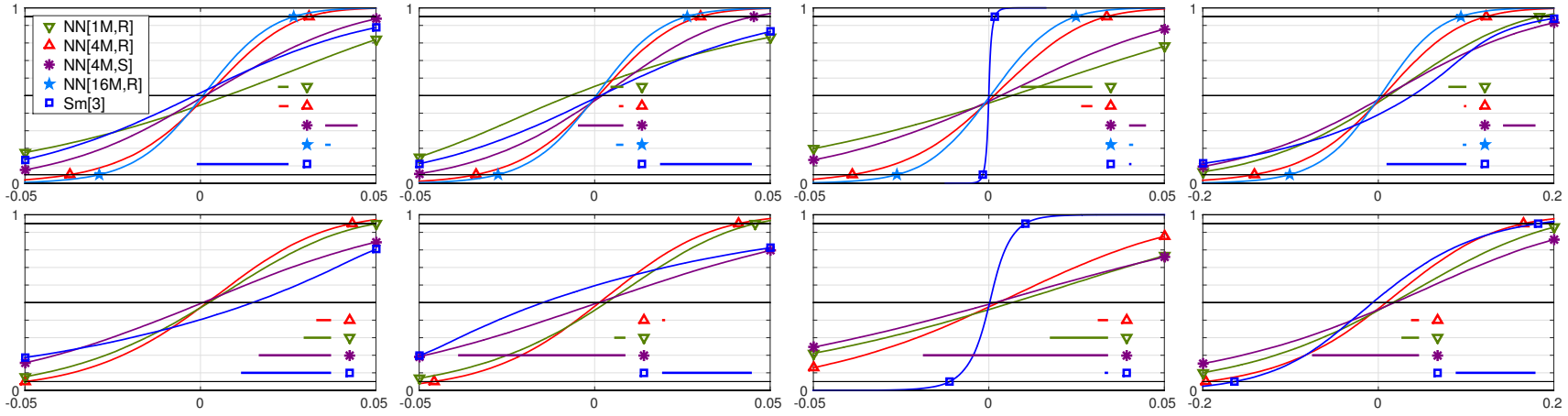


FIGURE 2. The empirical cumulative distribution function of the bias-free prediction errors for the eigenvalues $\#2, 3, 4, 5$ (left to right). The prediction bias is shown as the bar in the lower right corner. Training levels $L = 3$ (top) and $L = 2$ (bottom).

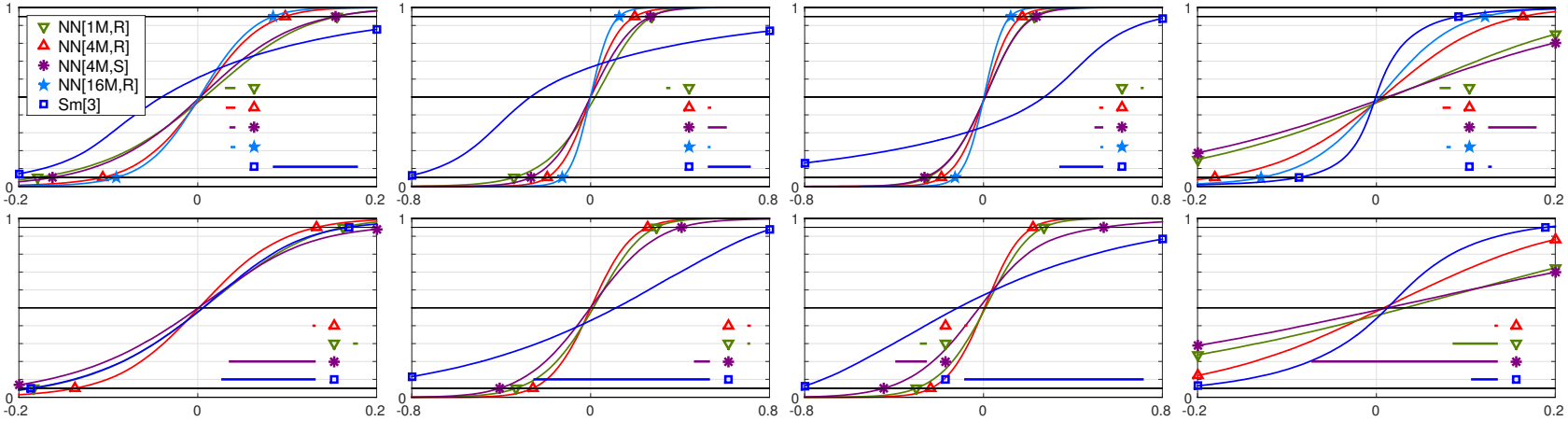


FIGURE 3. As in Figure 2 for the eigenvalues $\#6, 7, 8, 9$ (left to right).